November 15, 2006
Blender Py. Coding
(some edits made on Dec.26, 2006)

I should recap on what I have done these past nights. These days I make a special point of coding a bit each night. And it's not that hard. In fact, it's a bit relaxing and reassuring that I'm on the right track towards realization of some very cool things down the road.

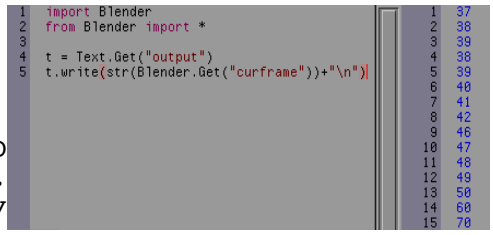
I take care to make sure the scripts are functioning correctly and not creating a massive hoard of random extra files or objects in my blender projects. I'm much more patient to do the job right than I ever was in the past. Ultimately, that should improve my efficiency in the future.

In the process I'm learning a tremendous deal about python. Just now, I found it could pass the def functions as parameters themselves. That was impressive. I'm sure I would never have know it had it not been for my programming languages courses. I guess the significance really hits you only once you know what to look for and how special little details like that can be.

... I had coded a few examples recently. Bits and pieces of stuff that will be put to work a lot better later. In the mean time I make sure to constantly be reading through that Python/Blender API. I salivate over the latest release of functions I have the privilege to use in version 2.42

I recap because I need to clean up my files and take better care of my hard drive space. This goes more for movies but the mind set has to be there all the time for me to bring anything to it's conclusion.

So where did I start?



It's safe to say I wanted to pick up where i left off last time I did some Blender Scripting. The last time was the addition of 3D text into the project via the code. Here, I began innocently enough by stepping back and managing and writing standard internal text files.

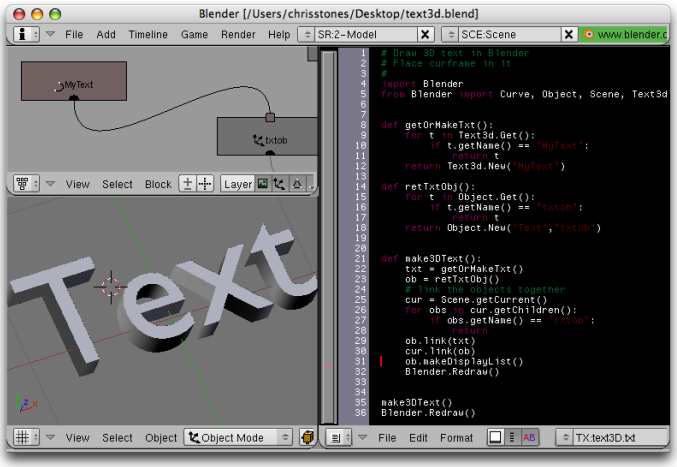
I liked the way I could get the output to register in the window as things were happening in the 3D window. It created the effect of something like a digital ticker tape that scrolled the values and of course they would remain when everything settled down. I knew I wanted a way to register output directly in some window while I was working so that I didn't have to keep checking back and forth between my console and blender.

```
108
1  import Blender
2  from Blender import *
3
4  def findOrMakeTxt():
5      for i in Text.Get():
6          if i.getName() == "outText":
7              return i
8      return Text.New("outText")
9      # make the file to show output for running progs
10     # else fiel outText exits so get
11
12 txt = findOrMakeTxt()
13
14 txt.clear()
15 txt.write( str(Blender.Get("curframe")) + " " )
```

I grew tired of the whole printing out everything and decided I wanted to make my new fabulous output window present only the formated output. I removed the scrolling numbers and replaced it with numbers updating in place.

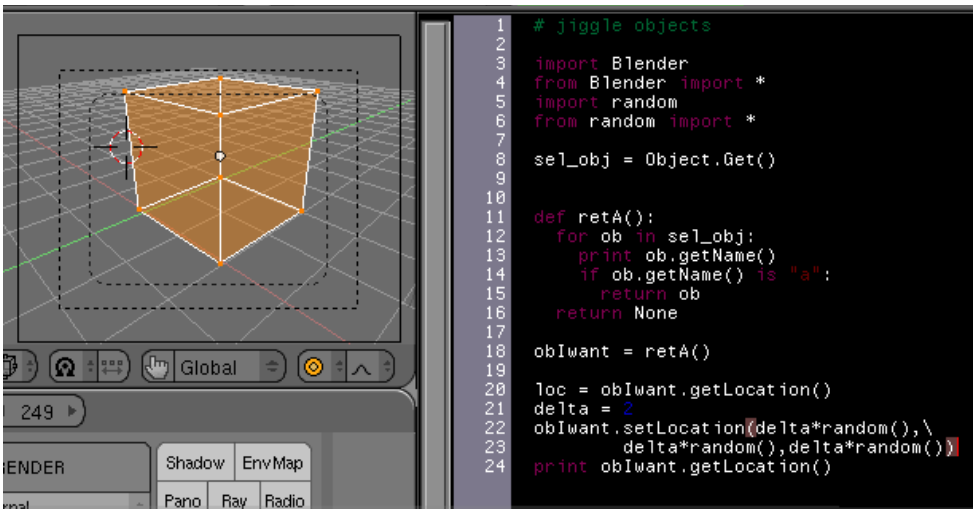
The code posed a bit of a challenge for me because I had to write it in such a way as to eliminate the constant copying of text files. I managed to solve the problem but I still don't know how elegant my solution is.

I had a reoccurring issue with duplicated objects and then problems selecting the same. I keep thinking there ha to be a better way to code it, but I'm at a loss.

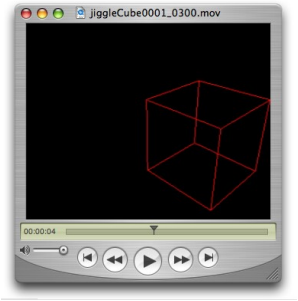


I moved on to make the text into actual 3D objects in the 3D Scene. This involved a lot of linking and error solving. Eventually I did it. And I was left with code that ran smoothly and didn't fill the scene with excess objects.

The reason I want text is because I want to show data directly in movies I render out of my projects. This is vital for future simulations. That's why there was so much focus on text first. Now that I had that under control I moved on a bit.



I moved on to some basic animations of objects in the scene. I randomly moved a cube a bit every frame so that when I rendered out to a movie it would jump around. It was a lovely simplistic demo that resulted in a red wire cube jittery movie.

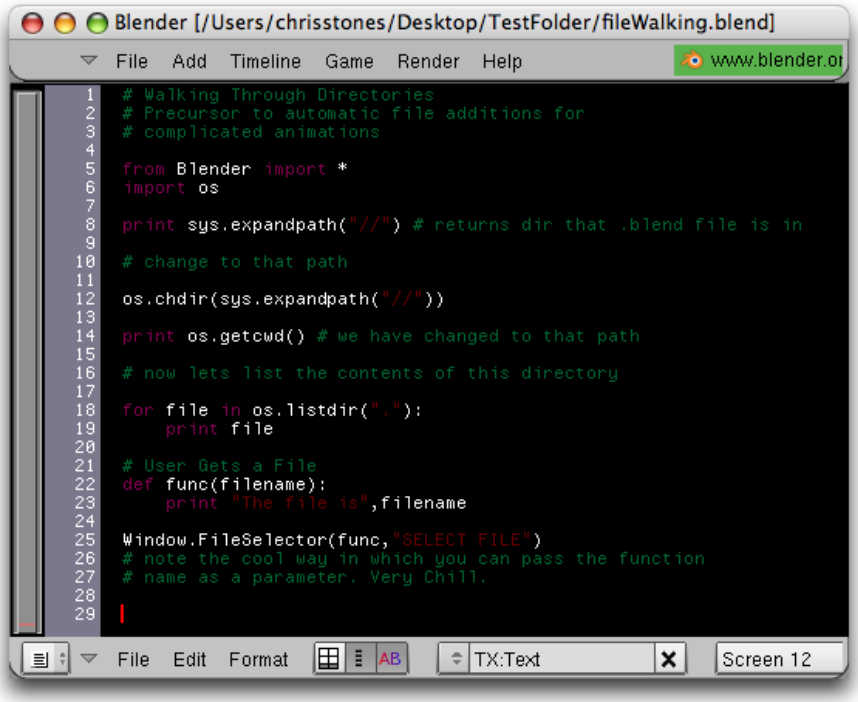


I purposely rendered in wire frame as well as turned off the line smoothing in order to get a ghetto old school digital effect. I have a wide range of options for control of how my simulations are going to look so I've been playing with different ideas. Wire frame is also the fastest and when combined with small dimension size it becomes a very good preview format.

This current night.. er more like earlier morning. I decided to make sure I could access the directories in order to eventually make some automatic parsing and animation scripts from folders of images. I've been dreaming of that for a while.

Ultimately, I feel that time spent learning this API and coding disciple is going to pay off much more than fibbing about in the standard blender manipulation channels. I want very complicated animations but I don't want to have to make them by hand. I want to be smarter than that and work out the programs to give me some very complicated effects.

Once I have these skills it shouldn't take so long.


Blender [/Users/chrisstones/Desktop/TestFolder/fileWalking.blend]
File Add Timeline Game Render Help
www.blender.or
# Walking Through Directories
# Precursor to automatic file additions for
# complicated animations
from Blender import *
import os
print sys.expandpath("//") # returns dir that .blend file is in
# change to that path
os.chdir(sys.expandpath("//"))
print os.getcwd() # we have changed to that path
# now lets list the contents of this directory
for file in os.listdir("."):
print file
# User Gets a File
def func(filename):
print "The file is",filename
Window.FileSelector(func,"SELECT FILE")
# note the cool way in which you can pass the function
# name as a parameter. Very Chill.
File Edit Format
AB
TX:Text
Screen 12